

# SPIDER-DAY01

## 1. 网络爬虫概述

- 1 【1】定义
- 2     1.1) 网络蜘蛛、网络机器人，抓取网络数据的程序
- 3     1.2) 其实就是用Python程序模仿人点击浏览器并访问网站，而且模仿的越逼真越好
- 4
- 5 【2】爬取数据的目的
- 6     2.1) 公司项目的测试数据，公司业务所需数据
- 7     2.2) 获取大量数据，用来做数据分析
- 8
- 9 【3】企业获取数据方式
- 10     3.1) 公司自有数据
- 11     3.2) 第三方数据平台购买(数据堂、贵阳大数据交易所)
- 12     3.3) 爬虫爬取数据
- 13
- 14 【4】Python做爬虫优势
- 15     4.1) Python：请求模块、解析模块丰富成熟，强大的Scrapy网络爬虫框架
- 16     4.2) PHP：对多线程、异步支持不太好
- 17     4.3) JAVA：代码笨重，代码量大
- 18     4.4) C/C++：虽然效率高，但是代码成型慢
- 19
- 20 【5】爬虫分类
- 21     5.1) 通用网络爬虫(搜索引擎使用，遵守robots协议)
- 22         robots协议：网站通过robots协议告诉搜索引擎哪些页面可以抓取，哪些页面不能抓取，通用网络爬虫需要遵守robots协议(君子协议)
- 23         示例：<https://www.baidu.com/robots.txt>
- 24     5.2) 聚焦网络爬虫：自己写的爬虫程序

## 2. 爬虫请求模块

### 2.1 requests 模块

#### ■ 安装

```
1  【1】 Linux
2      sudo pip3 install requests
3
4  【2】 Windows
5      方法1> cmd命令行 -> python -m pip install requests
6      方法2> 右键管理员进入cmd命令行 : pip install requests
```

## 2.2 常用方法

### ■ requests.get()

```
1  【1】 作用
2      向目标网站发起请求, 并获取响应对象
3
4  【2】 参数
5      2.1> url : 需要抓取的URL地址
6      2.2> headers : 请求头
7      2.3> timeout : 超时时间, 超过时间会抛出异常
```

### ■ 此生第一个爬虫

```
1  """
2  向京东官网 (https://www.jd.com/) 发请求, 获取响应内容
3  """
4  import requests
5
6  resp = requests.get(url='https://www.jd.com/')
7  # 1.text属性: 获取响应内容-字符串
8  html = resp.text
9  print(html)
```

### ■ 响应对象 (res) 属性

```
1  【1】 text      : 字符串
2  【2】 content   : 字节流
3  【3】 status_code : HTTP响应码
4  【4】 url       : 实际数据的URL地址
```

### ■ 重大问题思考

网站如何来判定是人类正常访问还是爬虫程序访问? --检查请求头!!!

```

1 # 请求头 (headers) 中的 User-Agent
2 # 测试案例: 向测试网站http://httpbin.org/get发请求, 查看请求头(User-Agent)
3 import requests
4
5 url = 'http://httpbin.org/get'
6 res = requests.get(url=url)
7 html = res.text
8 print(html)
9 # 请求头中:User-Agent为-> python-requests/2.22.0 那第一个被网站干掉的是谁??? 我们是不是需要
   # 发送请求时重构一下User-Agent??? 添加 headers 参数!!!

```

#### ■ 重大问题解决 - headers参数

```

1 """
2 包装好请求头后,向测试网站发请求,并验证
3 养成好习惯, 发送请求携带请求头, 重构User-Agent
4 """
5 import requests
6
7 url = 'http://httpbin.org/get'
8 headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.1 (KHTML,
   like Gecko) Chrome/14.0.835.163 Safari/535.1'}
9 html = requests.get(url=url,headers=headers).text
10 # 在html中确认User-Agent
11 print(html)

```

## 3. URL地址拼接

### 3.1 拼接URL地址的三种方式

```

1 【1】字符串相加
2 【2】字符串格式化 (占位符 %s)
3 【3】format()方法
4 'http://www.baidu.com/s?{}'.format(params)
5
6 【练习】
7 进入瓜子二手车直卖网官网 - 我要买车 - 请使用3种方法拼接前20页的URL地址,从终端打印输出
8 官网地址: https://www.guazi.com/langfang/
9
10 url = 'https://www.guazi.com/dachang/buy/o{}/#bread'
11 for i in range(1, 21):
12     page_url = url.format(i)
13     print(page_url)

```

### 3.2 练习

在百度中输入要搜索的内容, 把响应内容保存到本地文件

```

1  """
2  在百度中输入要搜索的内容，把响应内容保存到本地文件
3  """
4  import requests
5
6  # 1. 拼接URL地址
7  word = input('请输入搜索关键字:')
8  url = 'http://www.baidu.com/s?wd={}'.format(word)
9
10 # 2. 发请求获取响应内容
11 headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.1 (KHTML, like
12   Gecko) Chrome/14.0.835.163 Safari/535.1'}
13
14 # 3. 保存到本地文件
15 filename = '{}.html'.format(word)
16 with open(filename, 'w', encoding='utf-8') as f:
17     f.write(html)

```

## 4. 百度贴吧爬虫

### 4.1 需求

- 1 1、输入贴吧名称：赵丽颖吧
- 2 2、输入起始页：1
- 3 3、输入终止页：2
- 4 4、保存到本地文件：赵丽颖吧\_第1页.html、赵丽颖吧\_第2页.html

### 4.2 实现步骤

- 1 【1】查看所抓数据在响应内容中是否存在
- 2 右键 - 查看网页源码 - 搜索关键字
- 3
- 4 【2】查找并分析URL地址规律
- 5 第1页：http://tieba.baidu.com/f?kw=???&pn=0
- 6 第2页：http://tieba.baidu.com/f?kw=???&pn=50
- 7 第n页：pn=(n-1)\*50
- 8
- 9 【3】发请求获取响应内容
- 10
- 11 【4】保存到本地文件

### 4.3 代码实现

```

1  """
2      1、输入贴吧名称: 赵丽颖吧
3      2、输入起始页: 1
4      3、输入终止页: 2
5      4、保存到本地文件: 赵丽颖吧_第1页.html、赵丽颖吧_第2页.html
6  """
7  import requests
8  from urllib import parse
9  import time
10 import random
11
12 class TiebaSpider:
13     def __init__(self):
14         """定义常用变量"""
15         self.url = 'http://tieba.baidu.com/f?kw={}&pn={}'
16         self.headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.1 (KHTML, like Gecko) Chrome/14.0.835.163 Safari/535.1'}
17
18     def get_html(self, url):
19         """请求功能函数"""
20         html = requests.get(url=url, headers=self.headers).text
21
22         return html
23
24     def parse_html(self):
25         """解析功能函数"""
26         pass
27
28     def save_html(self, filename, html):
29         """数据处理函数"""
30         with open(filename, 'w') as f:
31             f.write(html)
32
33     def run(self):
34         """程序入口函数"""
35         name = input('请输入贴吧名:')
36         start = int(input('请输入起始页:'))
37         end = int(input('请输入终止页:'))
38         # 拼接多页的URL地址
39         for page in range(start, end + 1):
40             pn = (page - 1) * 50
41             page_url = self.url.format(name, pn)
42             # 请求 + 解析 + 数据处理
43             html = self.get_html(url=page_url)
44             filename = '{}_第{}页.html'.format(name, page)
45             self.save_html(filename, html)
46             # 终端提示
47             print('第%d页抓取完成' % page)
48             # 控制数据抓取的频率
49             time.sleep(random.randint(1, 2))
50
51 if __name__ == '__main__':
52     spider = TiebaSpider()
53     spider.run()

```

## 5. 正则解析模块re

### 5.1 使用流程

```
1 r_list=re.findall('正则表达式',html,re.S)
```

### 5.2 元字符

元字符	含义
.	任意一个字符（不包括\n）
\d	一个数字
\s	空白字符
\S	非空白字符
[]	包含[]内容
*	出现0次或多次
+	出现1次或多次

- 思考 - 匹配任意一个字符的正则表达式？

```
1 r_list = re.findall('.', html, re.S)
```

### 5.3 贪婪与非贪婪

- 贪婪匹配(默认)

```
1 1、在整个表达式匹配成功的前提下,尽可能多的匹配 * + ?  
2 2、表示方式: .*? .+? .??
```

- 非贪婪匹配

```
1 1、在整个表达式匹配成功的前提下,尽可能少的匹配 * + ?  
2 2、表示方式: .*? .+? .??
```

- 代码示例

```
1 import re  
2
```

```

3  html = '''
4  <div><p>九霄龙吟惊天变</p></div>
5  <div><p>风云际会潜水游</p></div>
6  '''
7  # 贪婪匹配
8  p = re.compile('<div><p>.*</p></div>', re.S)
9  r_list = p.findall(html)
10 print(r_list)
11
12 # 非贪婪匹配
13 p = re.compile('<div><p>.*?</p></div>', re.S)
14 r_list = p.findall(html)
15 print(r_list)

```

## 5.4 正则分组

### ■ 作用

1 在完整的模式中定义子模式，将每个圆括号中子模式匹配出来的结果提取出来

### ■ 示例代码

```

1  import re
2
3  s = 'A B C D'
4  r1 = re.findall('\w+\s+\w+', s, re.S)
5  print(r1)
6  # 分析结果是什么??
7  # 结果: ['A B', 'C D']
8
9  r2 = re.findall('(\w+)\s+\w+', s, re.S)
10 print(r2)
11 # 第一步: ['A B', 'C D']
12 # 第二步: ['A', 'C']
13
14 r3 = re.findall('(\w+)\s+(\w+)', s, re.S)
15 print(r3)
16 # 第一步: ['A B', 'C D']
17 # 第二步: [('A', 'B'), ('C', 'D')]

```

### ■ 分组总结

```

1  1、在网页中,想要什么内容,就加()
2  2、先按整体正则匹配,然后再提取分组()中的内容
3     如果有2个及以上分组(),则结果中以元组形式显示 [( ), ( ), ( )]
4  3、最终结果有3种情况
5     情况1: []
6     情况2: ['', '', ''] -- 正则中1个分组时
7     情况3: [( ), ( ), ( )] -- 正则中多个分组时

```

## ■ 课堂练习

```
1 # 从如下html代码结构中完成如下内容信息的提取:
2 问题1 : [('Tiger', ' Two...'), ('Rabbit', 'Small..')]
3 问题2 :
4     动物名称 : Tiger
5     动物描述 : Two tigers two tigers run fast
6     *****
7     动物名称 : Rabbit
8     动物描述 : Small white rabbit white and white
```

## ■ 页面结构如下

```
1 <div class="animal">
2     <p class="name">
3         <a title="Tiger"></a>
4     </p>
5     <p class="content">
6         Two tigers two tigers run fast
7     </p>
8 </div>
9
10 <div class="animal">
11     <p class="name">
12         <a title="Rabbit"></a>
13     </p>
14
15     <p class="content">
16         Small white rabbit white and white
17     </p>
18 </div>
```

## ■ 练习答案

```
1 import re
2
3 html = '''<div class="animal">
4     <p class="name">
5         <a title="Tiger"></a>
6     </p>
7
8     <p class="content">
9         Two tigers two tigers run fast
10    </p>
11 </div>
12
13 <div class="animal">
14     <p class="name">
15         <a title="Rabbit"></a>
16     </p>
17
18     <p class="content">
19         Small white rabbit white and white
20     </p>
21 </div>'''
```



```

22
23 p = re.compile('<div class="animal">.*?title="(.*?)".*?content">(.*?)</p>.*?
    </div>',re.S)
24 r_list = p.findall(html)
25
26 for rt in r_list:
27     print('动物名称:',rt[0].strip())
28     print('动物描述:',rt[1].strip())
29     print('*' * 50)

```

## 6. 笔趣阁小说爬虫

### 6.1 项目需求

- 1 【1】 官网地址: <https://www.biqukan.cc/list/>
- 2 选择一个类别, 比如: '玄幻小说'
- 3
- 4 【2】 爬取目标
- 5 '玄幻小说'类别下前20页的
- 6 2.1》小说名称
- 7 2.2》小说链接
- 8 2.3》小说作者
- 9 2.4》小说描述

### 6.2 思路流程

- 1 【1】 查看网页源码, 确认数据来源
- 2 响应内容中存在所需抓取数据
- 3
- 4 【2】 翻页寻找URL地址规律
- 5 第1页: <https://www.biqukan.cc/fenlei1/1.html>
- 6 第2页: <https://www.biqukan.cc/fenlei1/2.html>
- 7 第n页: <https://www.biqukan.cc/fenlei1/n.html>
- 8
- 9 【3】 编写正则表达式
- 10 '<div class="caption">.\*?<a href="(.\*?)" title="(.\*?)">.\*?<small.\*?>(.\*?)</small>.\*?>(.\*?)</p>'
- 11
- 12 【4】 开干吧兄弟

### 6.3 代码实现

```

1 """
2 目标:

```

```

3      笔趣阁玄幻小说数据抓取
4 思路:
5      1. 确认数据来源 - 右键 查看网页源代码,搜索关键字
6      2. 确认静态,观察URL地址规律
7      3. 写正则表达式
8      4. 写代码
9      """
10
11     import re
12     import requests
13     import time
14     import random
15
16     class NovelSpider:
17         def __init__(self):
18             self.url = 'https://www.biqukan.cc/fenlei1/{}.html'
19             self.headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; WOW64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.193 Safari/537.36'}
20
21         def get_html(self, url):
22             html = requests.get(url=url, headers=self.headers).content.decode('gbk', 'ignore')
23
24             self.refunc(html)
25
26         def refunc(self, html):
27             """正则解析函数"""
28             regex = '<div class="caption">.*?<a href="(.*?)" title="(.*?)">.*?<small.*?>(.*?)
</small>.*?>(.*?)</p>'
29             novel_info_list = re.findall(regex, html, re.S)
30             for one_novel_info_tuple in novel_info_list:
31                 item = {}
32                 item['title'] = one_novel_info_tuple[1].strip()
33                 item['href'] = one_novel_info_tuple[0].strip()
34                 item['author'] = one_novel_info_tuple[2].strip()
35                 item['comment'] = one_novel_info_tuple[3].strip()
36                 print(item)
37
38         def crawl(self):
39             for page in range(1, 6):
40                 page_url = self.url.format(page)
41                 self.get_html(url=page_url)
42                 time.sleep(random.randint(1, 2))
43
44     if __name__ == '__main__':
45         spider = NovelSpider()
46         spider.crawl()

```

## 7. MySQL数据持久化

### 7.1 pymysql回顾

#### ■ MySQL建库建表

```

1 create database noveldb charset utf8;
2 use noveldb;
3 create table novel_tab(
4 title varchar(100),
5 href varchar(500),
6 author varchar(100),
7 comment varchar(500)
8 )charset=utf8;

```

#### ▪ pymysql示例

```

1 import pymysql
2
3 db = pymysql.connect('localhost','root','123456','noveldb',charset='utf8')
4 cursor = db.cursor()
5
6 ins = 'insert into novel_tab values(%s,%s,%s,%s)'
7 novel_li = ['花千骨', 'http://zly.com', '赵丽颖', '小骨的传奇一生']
8 cursor.execute(ins,novel_li)
9
10 db.commit()
11 cursor.close()
12 db.close()

```

## 7.2 笔趣阁数据持久化

```

1 """
2 1. 在 __init__() 中连接数据库并创建游标对象
3 2. 在数据处理函数中将所抓取的数据处理成列表，使用execute()方法写入数据库
4 3. 数据抓取完成后关闭游标及断开数据库连接
5 """
6 import re
7 import requests
8 import time
9 import random
10 import pymysql
11
12 class NovelSpider:
13     def __init__(self):
14         self.url = 'https://www.biqukan.cc/fenlei1/{ }.html'
15         self.headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; WOW64)
16 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.193 Safari/537.36'}
17         # 连接数据库
18         self.db = pymysql.connect('localhost', 'root', '123456', 'noveldb', charset='utf8')
19         self.cur = self.db.cursor()
20
21     def get_html(self, url):
22         html = requests.get(url=url, headers=self.headers).content.decode('gbk', 'ignore')
23
24         self.refunc(html)

```

```

25     def refunc(self, html):
26         """正则解析函数"""
27         regex = '<div class="caption">.*?<a href="(.*?)" title="(.*?)">.*?<small.*?>(.*?)</small>.*?>(.*?)</p>'
28         novel_info_list = re.findall(regex, html, re.S)
29         for one_novel_info in novel_info_list:
30             # 调用数据处理函数
31             self.save_to_mysql(one_novel_info)
32
33     def save_to_mysql(self, one_novel_info):
34         """将数据存入MySQL数据库"""
35         one_novel_li = [
36             one_novel_info[1].strip(),
37             one_novel_info[0].strip(),
38             one_novel_info[2].strip(),
39             one_novel_info[3].strip(),
40         ]
41         ins = 'insert into novel_tab values(%s,%s,%s,%s)'
42         self.cur.execute(ins, one_novel_li)
43         self.db.commit()
44         # 终端打印测试
45         print(one_novel_li)
46
47     def crawl(self):
48         for page in range(1, 6):
49             page_url = self.url.format(page)
50             self.get_html(url=page_url)
51             time.sleep(random.randint(1, 2))
52
53         # 所有数据抓取完成后断开数据库连接
54         self.cur.close()
55         self.db.close()
56
57 if __name__ == '__main__':
58     spider = NovelSpider()
59     spider.crawl()

```

## 8. 今日作业

```
1  【1】把百度贴吧案例重写一遍,不要参照课上代码
2  【2】笔趣阁案例重写一遍,不要参照课上代码
3  【3】复习任务
4      pymysql、MySQL基本命令
5      MySQL  : 建库建表普通查询、插入、删除等
6      Redis  :  python和redis交互,集合基本操作
7  【4】猫眼电影top100数据抓取
8      url地址: https://maoyan.com/board/4
9      所抓数据: 电影名称
10             电影主演
11             上映时间
12      数据处理: 每个电影作为字典打印输出
13      # 因频率过高出现滑块验证, 则在页面中手动滑动滑块通过验证
```